

Programme: **B. Tech.** Discipline: **CSE**

Semester **VI** Academic Year **2020**

**Mid-Sem 2** **Assignment**

Course Code: **CS4111** Course Title **Computer Vision**

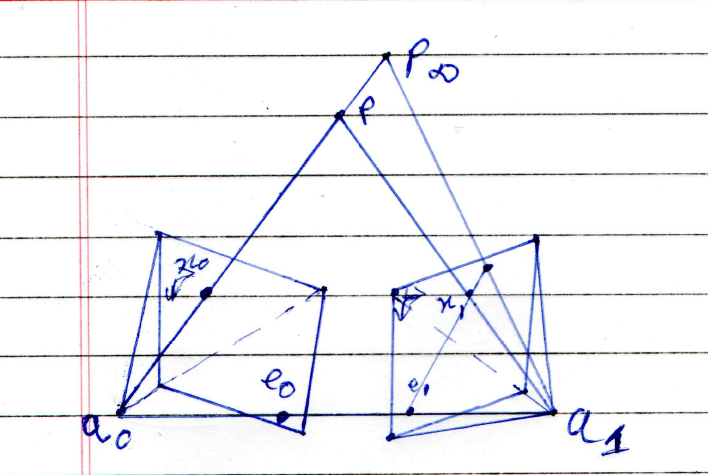
Date of Submission: **30/06/2020** Time: **11.55 PM**   
 Max. Marks: **15**

**Instructions**:

1. **Answer all questions. All questions carry equal marks.**
2. **You can write your answers in this document only by creating space after the questions.**
3. **Write your answers in bold fonts with a line spacing of 1.15.**
4. **This is an open book assignment. However, you should write your answers in your own words.**
5. **Do not copy the answers either from internet or from each other. If the answers are found to be copied, you would simply be awarded zero.**
6. **Do not share your answers with others. If the answers are found to be same no investigations would be done as to who has copied from whom. Both will be awarded zero.**
7. **If you wish to support your answers by diagrams/figures, either draw them using ‘insert shapes’ option in MS Word or you can draw using pen/pencil on a piece of paper, convert it into an image by taking snapshot with your mobile and paste in between your answers. Do not copy paste from internet/textbook/notes.**
8. **Although there is no word limit for your answers, but try to be logical and to the point while expressing your answers.**
9. **You have to submit you assignment on Moodle by 30th June (Tuesday) before 11.55 PM. No further extension will be given to you to submit.**
10. **No assignment will be accepted on mail.**
11. **It is advised to all of you to finish and submit your assignment well before time in order to avoid any technical glitches at last minute. I repeat, no assignment will be accepted on mail.**
12. **There is no makeup for this assignment.**
13. How can you compute the correspondence of a pixel, which is present in one image, in another image? Explain in detail.

**Ans) Correspondence is matching pixels of one image with those of another image.**

**A pixel (x0) in one image projects to an epipolar line segment in the other image.The segment is bounded at one end by the projection of the original viewing ray at infinity (p infinite in the diagram) and the other end by the projection of the original camera center (a0) into the second camera which is the epipole (e1). On projecting the epipolar line in the 2nd camera image back to the 1st we get another line this time bound by e0. In the diagram triangle (a0)(p)(a1) is the epipolar plane.**



**Extending both lines to infinity we get a pair of corresponding epipolar lines,which are the intersection of the two image planes with the epipolar plane that passes through both camera centers c0 and c1 as well as the point of interest (p).**

**We then use the epipolar line corresponding to the pixel in one image to constrain the search for corresponding pixels in the other image. After warping the input images the horizontal scan lines become the epipolar lines. With this we match the scanlines to compute correspondence.This is one of the methods to compute correspondence. We can also use sparse correspondence or dense correspondence.**

1. How can you extract texture feature using filter banks? What are the advantages of this technique? Explain in detail.

**Ans) In texture features we don’t know exactly what to look for as there is no defined set of fundamental micro-structures. Hence for texture we look for simple patterns at the dots and bars level. Such patterns are rather easily picked when we filter the image. Using convolution between the image and linear filters gives us a different representation of the image. This process makes the inherent structure of the image more recognisable as strong similarity between image pattern and filter pattern and hence gets more outlined while dissimilar parts of the image are not outlined as much.**

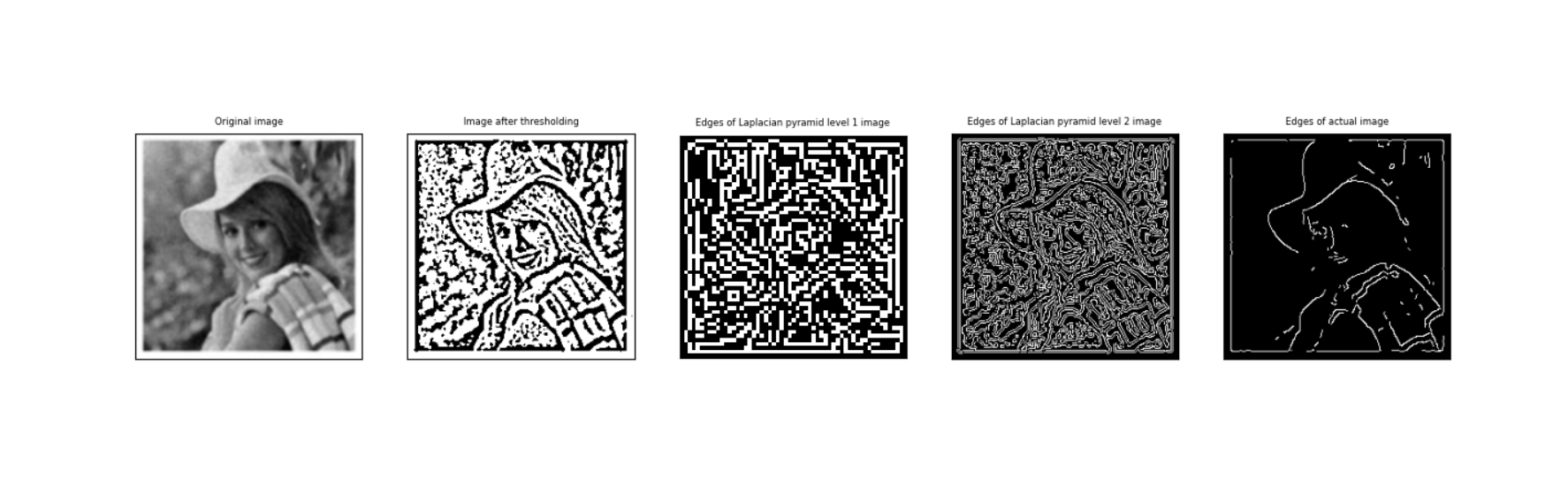
**A Filter bank can be said as a collection of digital filters, with either a common input or a common output. The purpose of the filter bank is usually to decompose the signal in subbands. An example is the Gaussian pyramid in which subsequent images are smoothed using a smoothing filter and then subsampling the smoothed image. This image is then subjected to the same procedure multiple times.**

**Advantage:-**

**1)Such representation, although redundant, reveals the structure in a useful way. From this texture analysis we can recover some shape information too. Shape from texture algorithms take a few steps, including the extraction of repeated patterns or the measurement of local frequencies in order to compute local affine deformations, and a subsequent stage to infer local surface orientation of the object.**

**2) After only 2 or 3 repetitions of the Gaussian pyramid (example) we get a very smooth output on which texture analysis gets a very good output.**

**3) It is easy to recover an image from its Laplacian pyramid.**

**[1)original image. 2)image after adaptive thresholding(to remove noise) 3)Edges of Laplacian pyramid level 1 image 4) Edges of Laplacian pyramid level 2 image 5) Edges of actual image(code for this is mentioned at the bottom of the assignment)]**

**Here we can clearly see that the edges, shape and textures of the image are better highlighted after being filtered (Laplacian pyramid in this case) than the edges of the actual image.**

1. Suppose there is an image containing multiple objects of different types. The objective of the problem is to recognize objects using primary features of objects (colour, texture, shape). Explain your own idea what concepts will you apply, and how will you apply them, to differentiate/classify the objects in the image by extracting primary features (or combination of features) of objects. Also, justify how your idea can produce the best accuracy.

**Ans) The best methodology will be a combination of all primary features so as to extract the most amount of information present in the image.**

**Because images can be highly cluttered and similar features may belong to several objects.The original set of feature matches can have a large number of outliers to deal with this; we can use Hough transform to accumulate votes for likely geometric transformations to use on the image so as to decrease the effect of such outliers.**

**Smoothing the image can prove to be helpful in case of noisy image dataset .Then we extract features at a variety of scales by performing the same operations at multiple resolutions in a pyramid and then matching those features at the same level.**

**For shape we can use local thresholding and edge linking to make then the outlines of the image. We can also use complete segmentation here if provided the resources. Regarding separation of objects region based segmentation can be helpful and is probably the best method.**

**Moment values and features can differentiate among objects of different shape. And if shape classes are labelled with landmarks on the images in the database then the recognition will be much simpler as we search only with objects of similar moment features.**

**The recognition system extracts a set of interest points in each database image and stores the associated descriptors (and original positions) in an indexing structure such as a search tree . At recognition time, the features that were previously extracted from the image are compared against the stored object features. Whenever a sufficient number of matching features are found for an object, the system returns the object which is most likely.**

**This process should be an efficient way for object detection as:-**

* **Most data that will be stored will not be the actual images, only the values of their texture, shape, colour analysis ; the processing faster than processing directly on images.**
* **As most of the algorithms mentioned above are somewhat simpler to understand, further modifications and improvement can be applied easily.**
* **As we use Hough transform, the effect of outliers is reduced.**
* **As we extract features at a variety of scales the program is suitable when the objects being matched do not undergo large scale changes.**
* **As most parts are not input image data it is light to save the model and directly reuse later.**

CODE REFERRED IN QUESTION -2

import cv2

import warnings

warnings.filterwarnings("ignore")

import matplotlib.pyplot as plt

for x in range(0,5):

image = cv2.imread(f'{x}.jpg',0)

try:

img = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

except:

img = image # incase image is already gray scale

img = cv2.GaussianBlur(img, (5,5),0)

threshimg = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C, cv2.THRESH\_BINARY,11,2)

L=[img,threshimg]

gaussianlist = [threshimg]

lower=threshimg

for i in range(2):

lower = cv2.pyrDown(lower)

gaussianlist.append(lower)

laplacian\_top = gaussianlist[-1]

laplaclist = [laplacian\_top]

for i in range(1, 0, -1):

size = (gaussianlist[i - 1].shape[1], gaussianlist[i - 1].shape[0])

gaussian\_exp = cv2.pyrUp(gaussianlist[i], dstsize=size)

lapla = cv2.subtract(gaussianlist[i - 1], gaussian\_exp)

laplaclist.append(lapla)

moment=[]

for i in range(len(laplaclist)):

edges = cv2.Canny(laplaclist[i], 149, 206)

L.append(edges)

moment.append(cv2.moments(edges))

L.append(cv2.Canny(img, 180, 200))

for i in range(len(L)):

plt.subplot(1,len(L),i+1), plt.imshow(L[i], 'gray')

plt.xticks([]),plt.yticks([])

plt.subplot(1,len(L),1).set\_title('Original image',fontsize = 6)

plt.subplot(1, len(L), 2).set\_title('Image after thresholding', fontsize=6)

plt.subplot(1, len(L), 3).set\_title('Edges of Laplacian pyramid level 1 image', fontsize=6)

plt.subplot(1, len(L), 4).set\_title('Edges of Laplacian pyramid level 2 image', fontsize=6)

plt.subplot(1, len(L), 5).set\_title('Edges of actual image', fontsize=6)

plt.show()